EXAMPLE NAME: XMC4500_Relax_Kit_DMA_FCE

OVERVIEW:

This example shows the usage of the DMA_CH APP. The DMA_CH APP it is used to transfer data between memory to memory, memory to peripheral, peripheral to memory or peripheral to peripheral without the intervention of the CPU.

DESCRIPTION:

In this example, the DMA_CH APP is used to feed the Flexible CRC Engine (FCE) to calculate the CRC-32 on a fictitious frame of data.

The CRC-32 of a frame of 256 words is calculated. If the calculated CRC matches the expected one, then the LED1 will be turned on, otherwise the LED1 will blink.

The example shows how to combine DAVE APPs and Peripheral Low Level Driver. In this case the LLD is used to configure the FCE.

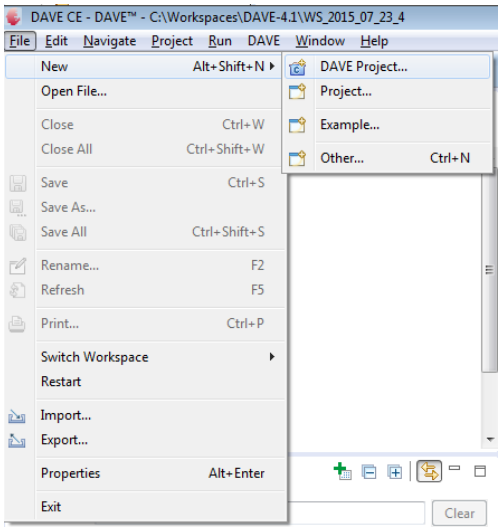
REQUIRMENTS:

Boards Required: XMC4500 Relex Kit Board (Order Nr: KIT_XMC45_RELAX_V1)

HOW TO CREATE THE PROJECT:

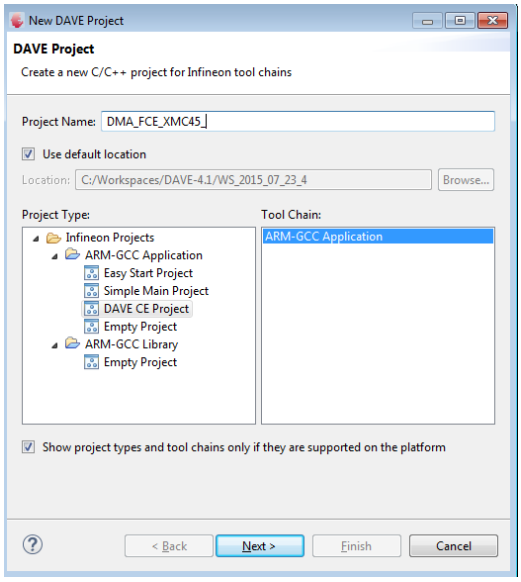
1. Create a new DAVE CE Project

   To create new projects go to **File** menu, select **New** and choose **DAVE Project**.



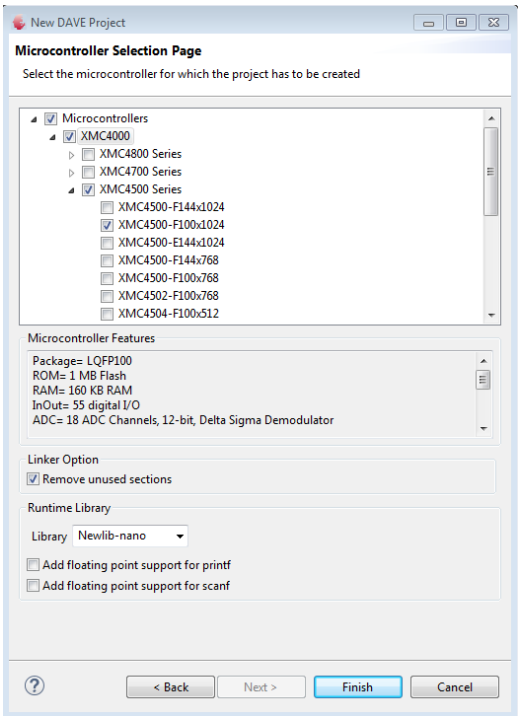
In the **New DAVE Project** window:

- in the *Project name:* field enter the name of the new project, i.e. XMC4500_Relax_Kit_External_Interrupt.
- in the *Project type:* section select **DAVE CE Project**.
- in the *Toolchains:* section select **ARM-GCC Application**
- click the Next > button



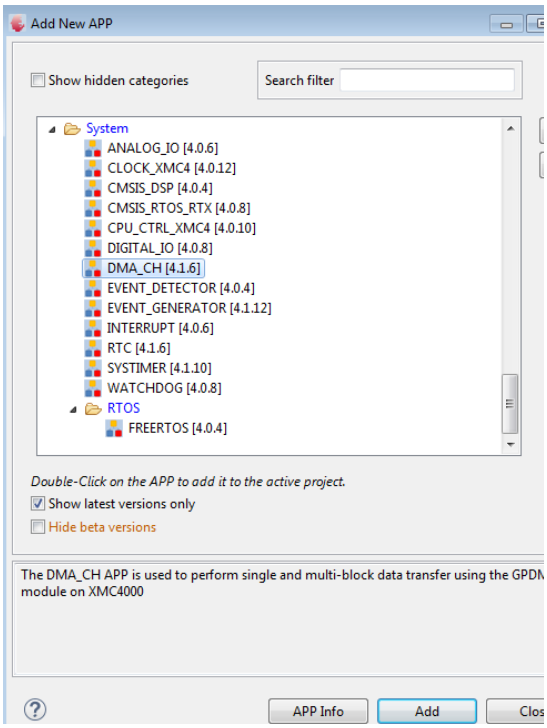
In the **Target Selection Page**:

- select the target device, i.e. **XMC4500-F100x1024**
- click the **Finish** button



2. Add APPs to your project

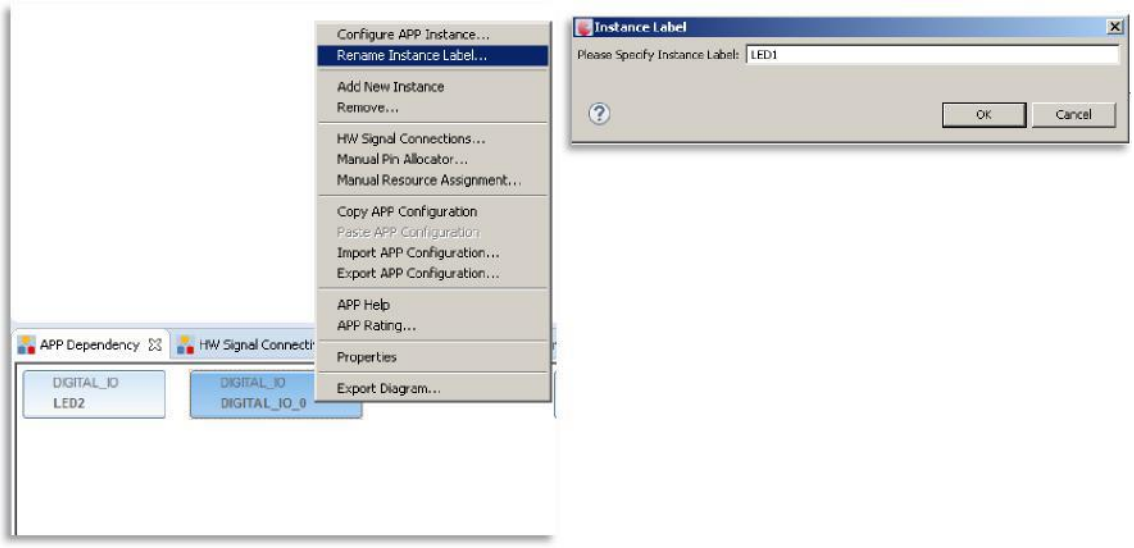
Use the *Add New APP* button in toolbar or go to **DAVE** menu and select *Add New APP*. In the **Add New APP** dialog, add the following APPs by double clicking:



- add a DMA_CH APP
- add a DIGITAL_IO APPs
- click the **Close** button

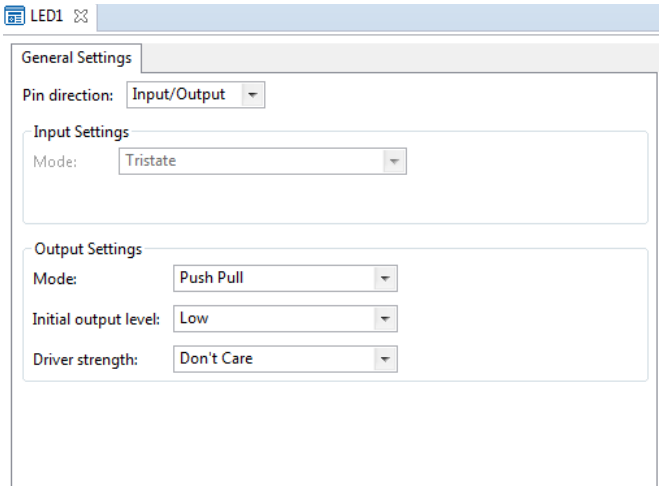
3. Assign names to APP instances

- Right click on the DIGITAL_IO_0 APP instance, select *Rename Instance Label...* and set the name to LED1.

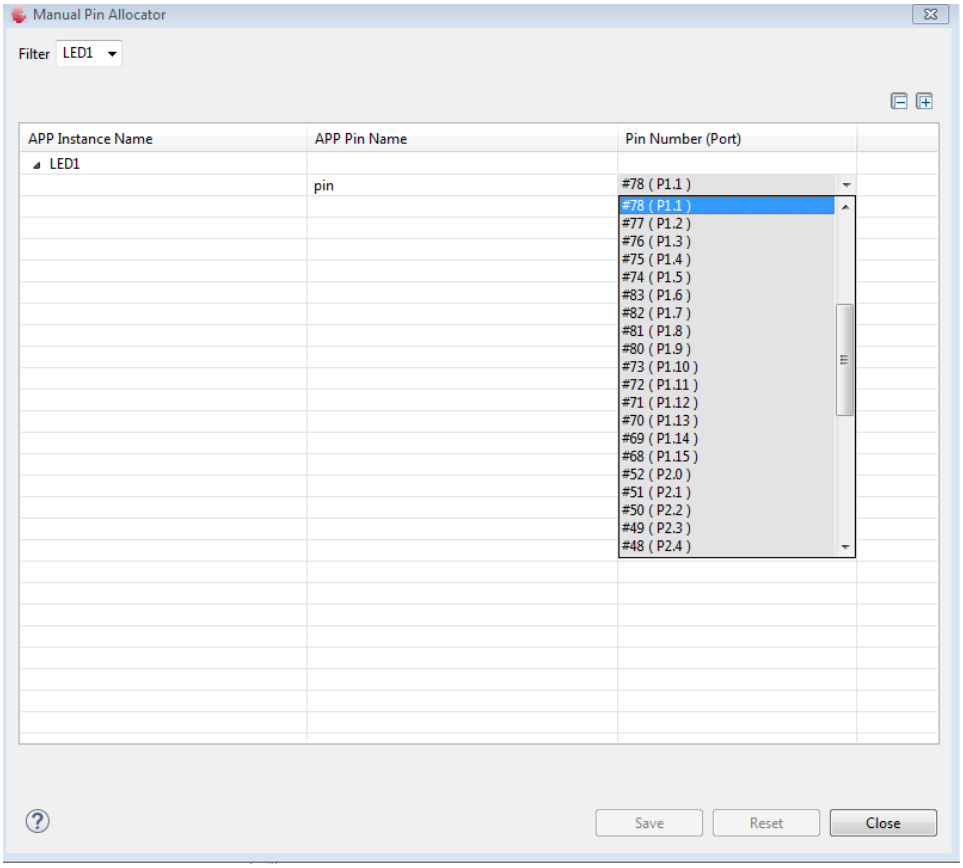

Configure APP Instance...
Rename Instance Label...
Add New Instance
Remove...
HW Signal Connections...
Manual Pin Allocator...
Manual Resource Assignment...
Copy APP Configuration
Paste APP Configuration
Import APP Configuration...
Export APP Configuration...
APP Help
APP Rating...
Properties
Export Diagram...
APP Dependency
HW Signal Connecti
DIGITAL_IO
LED2
DIGITAL_IO
DIGITAL_IO_0
Instance Label
Please Specify Instance Label:
LED1
OK
Cancel

4. Configure APPs and signals

- Double click on LED1 APP instance to configure the APP:
    - select **Input/Output** in *Pin Direction:* selection
    - keep default for the other of configuration settings.



- Right click on LED1 APP instance, select *Manual Pin Allocator...*
  In the **Manual Pin Allocator** dialog:
    - select P1.1 in the *Pin Number (Port)* selection
    - click on **Save** button and finally on **Close** button.


Manual Pin Allocator
Filter
LED1
APP Instance Name
APP Pin Name
Pin Number (Port)
LED1
pin
#78 (P1.1)
#78 (P1.1)
#77 (P1.2)
#76 (P1.3)
#75 (P1.4)
#74 (P1.5)
#83 (P1.6)
#82 (P1.7)
#81 (P1.8)
#80 (P1.9)
#73 (P1.10)
#72 (P1.11)
#71 (P1.12)
#70 (P1.13)
#69 (P1.14)
#68 (P1.15)
#52 (P2.0)
#51 (P2.1)
#50 (P2.2)
#49 (P2.3)
#48 (P2.4)
Save
Reset
Close

- Double click on DMA_CH_0 APP instance to configure the APP:

  In the Channel Settings tab:

  - select **Single block** in *Transfer type:* selection

    In this example a single block of data is transferred. After the transfer is complete the DMA channel will be disabled
  - select **Memory to memory (DMA flow controller)** in *Transfer flow:* selection Memory to memory is to be chosen when no handshaking is required. The FCE does not require handshaking, is always ready to accept new data.
  - select **Priority 0 (Lower)** in *Prioriry:* selection

    No other DMA channel is configured in this example, therefore is the priority is not relevant
  - enter **256** in *Block Size:*

    This is the size of the data transfer which matches the size of the frame
  - enter **src_ptr** in *Source Address:*

    This is the start of the data to be transferred. The content of the pointer need to be initialized by the user in the code
  - select **Increment** in *Source Address counter:* selection

    The internal DMA source address pointer is to be incremented after every single word transfer
  - select **32 bits** in *Source Transfer width:* selection

    This yields the best performance when transferring data from memory.
  - select **8 words** in *Source Burst width:* selection

    DMA0 CH0 and CH1 have a FIFO of 8 words. By this setting we can store a complete burst in the DMA and the bus utilization is minimized. The DAVE solver will use this information to assign the correct hardware resource to the DMA_CH APP, i.e. DMA0 CH0 or CH1.
  - enter **dst_ptr** in *Destination Address:*

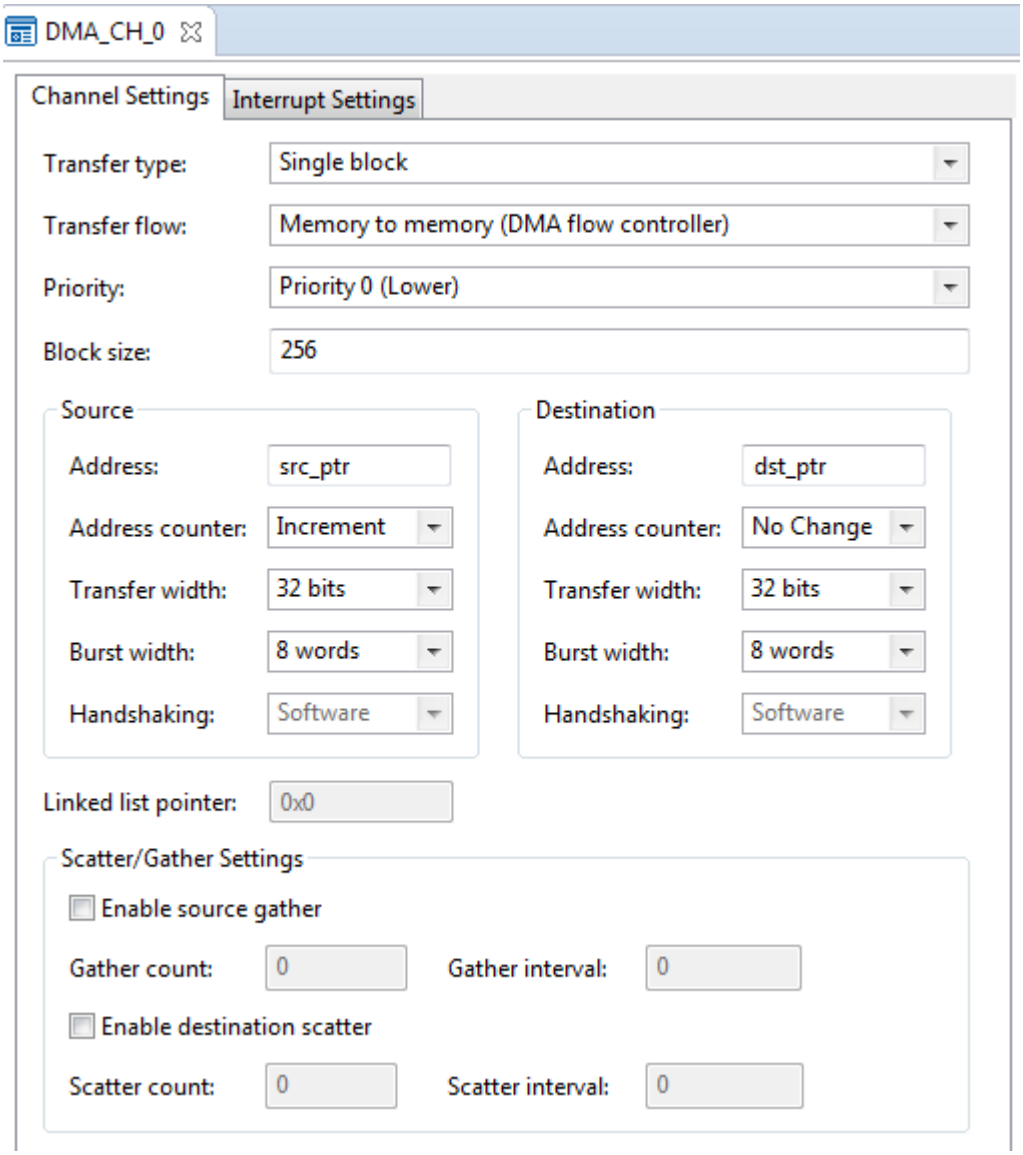
    pointer to the data register of the FCE
  - select **No Change** in *Destination Address counter:* selection

    we want to transfer always to the data register of the FCE
  - select **32 bits** in *Destination Transfer width:* selection

    The FCE data register only accepts writes of 32bits

- select **8 words** in *Source Burst width:* selection

  DMA0 CH0 and CH1 have a FIFO of 8 words. By this setting we can store a complete burst in the DMA and the bus utilization is minimized. The DAVE solver will use this information to assign the correct hardware resource to the DMA_CH APP, i.e. DMA0 CH0 or CH1.



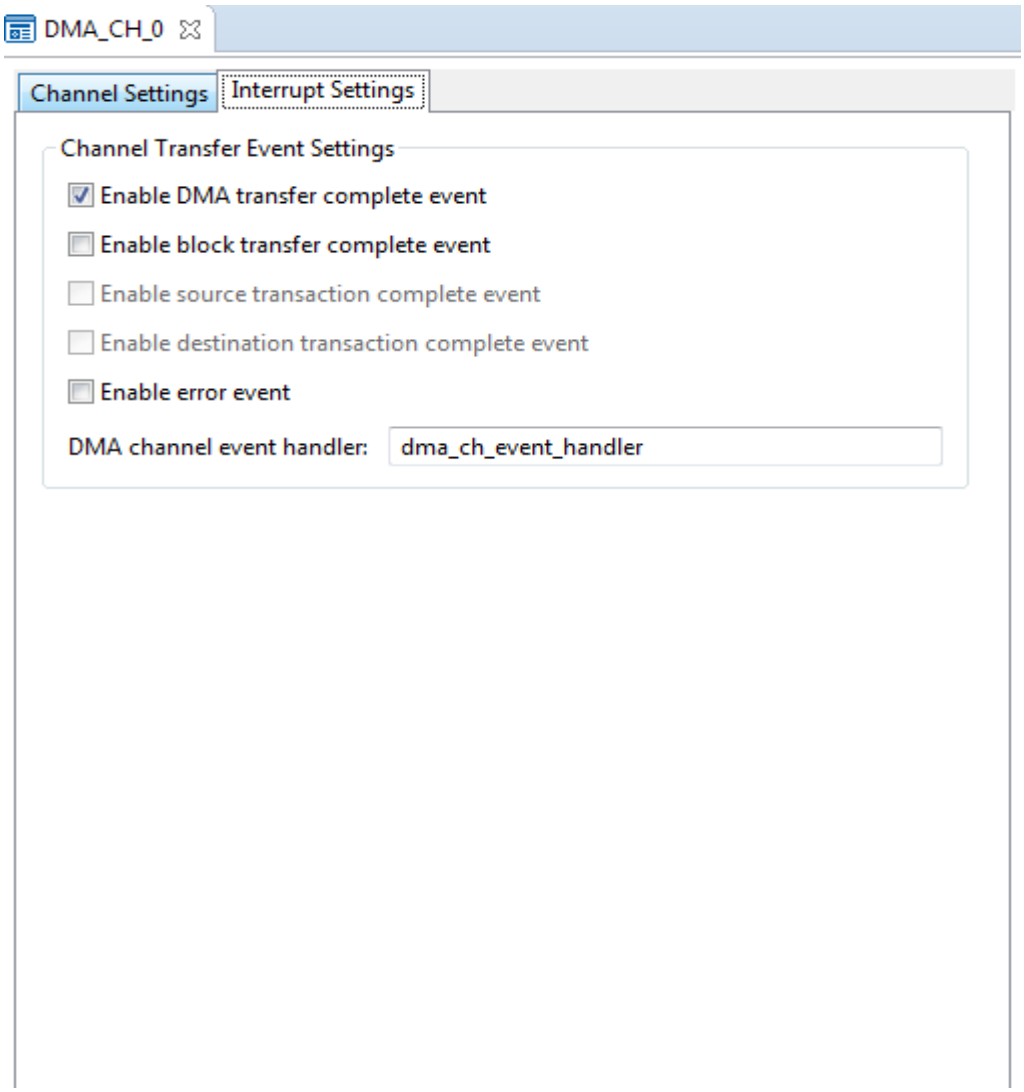
In the Event Settings tab:

- enable DMA transfer complete event

  The user will be notified by an interrupt when the DMA transfer is finished

- Rename the event handler to dma_ch_event_handler

  This the user call back that gets invoked when the interrupt happens. The user can customize how to react depending on the event causing the interrupt



5. Generate the code

Use the button  in toolbar or go to **DAVE** menu and select *Generate Code.*

In the project a new folder *Generated* is created under *DAVE* folder. This folder contains a subfolder for every APP type used in the project. Each APP subfolder contains the generated APP

configuration in the form of configuration structures, i.e. DMA_CH_conf.c. In addition, the APP API library files are copied.

6. Add user code to main.c (in parenthesis is the line number)
   (2) Add include file to use the FCE low level driver
   (5-39) Add the frame data
   (42-43) The pointers named in the UI are initialized
   (46-53) FCE configuration
   (59-78) Implementation of the user call back
   (104-107) The FCE is configured and initialized
   (110) The DMA CH is enabled, starting the immediately the data transfer, since no handshaking is has been selected.

7. Build and download to the microcontroller.